

TRAVAUX PRATIQUES - E.D.O.

Groupes B1 à B6

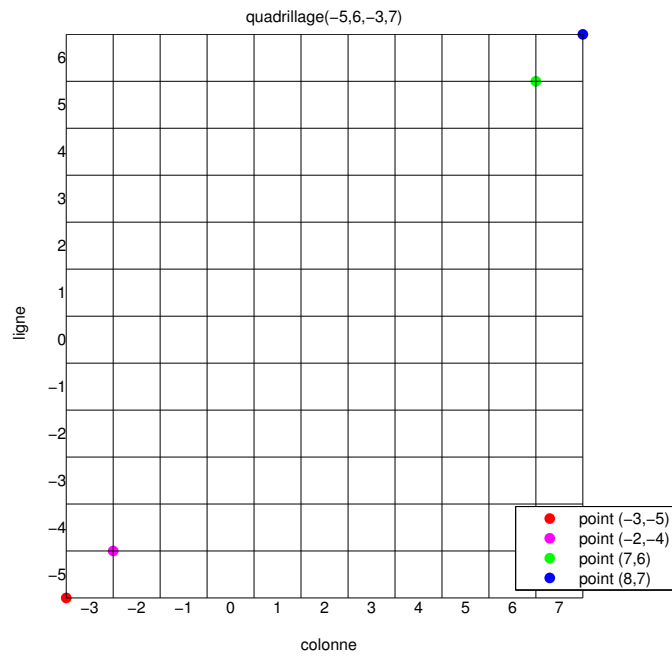
Travail individuel et personnel

Table des matières

1	Tests Algorithmique et Matlab	1
2	Résolution numérique d'équations différentielles ordinaires	4
2.1	Rappels de schémas numériques usuels	4
2.1.1	Schéma d'Euler progressif	4
2.1.2	Schéma de la tangente améliorée	5
2.1.3	Schéma de Runge-Kutta d'ordre 4	5
2.1.4	Méthodes d'Adams-Bashforth	5
2.1.5	Méthodes d'Adams-Moulton	5
2.2	Schéma prédicteur-correcteur	5
2.3	Travail à effectuer	6
3	Le système solaire	8
3.1	Position du problème et équations différentielles	8
3.2	Résolution numérique classique	8
3.3	Système à 3 corps : Soleil, Saturne et Jupiter	9
3.4	Système à 6 corps	9
3.5	Système à 7 corps	11
4	Annexes	11
4.1	Quelques E.D.O. du premier ordre	11
4.1.1	Exemple 1	11
4.1.2	Exemple 2	11
4.1.3	Exemple 3	11

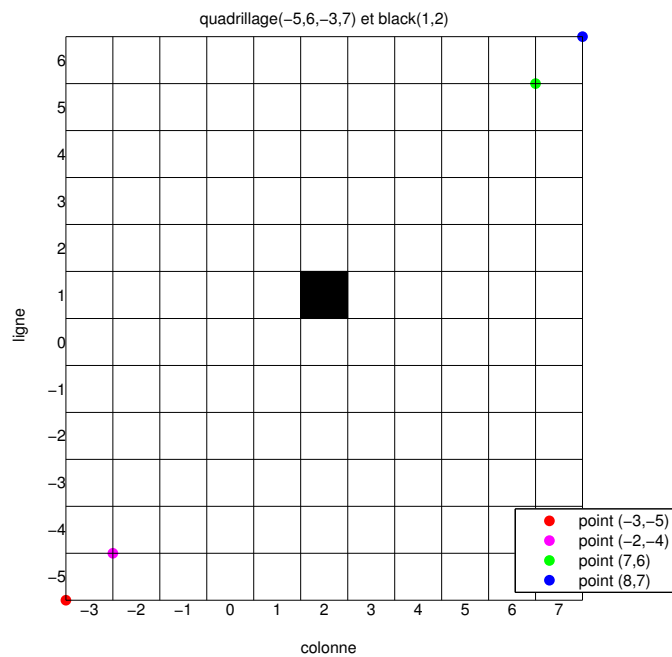
1 Tests Algorithmique et Matlab

Q. 1 *Ecrire la fonction Matlab `Quadrillage(imin,imax,jmin,jmax)` permettant de générer un quadrillage (i.e. uniquement les pour les lignes $imin$ à $imax$ et les colonnes $jmin$ à $jmax$. Voici un exemple avec la commande `Quadrillage(-5,6,-3,7)` représentant uniquement les traits noirs sur la figure :*

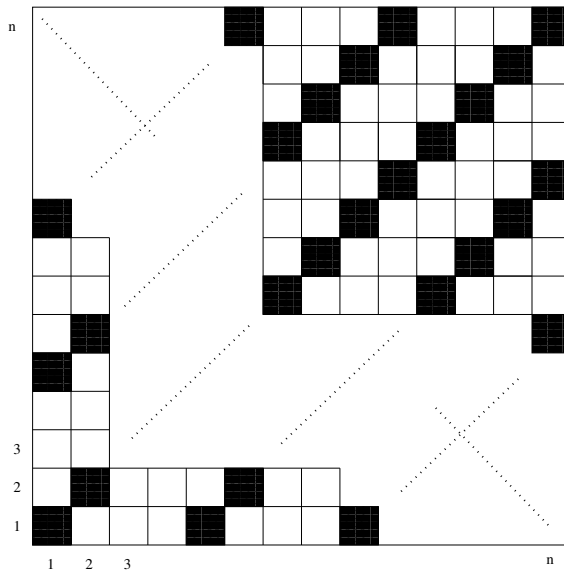


On peut tester cette fonction avec le programme *Quadrillagefigure* fourni pour obtenir la figure précédente et la suivante. ■

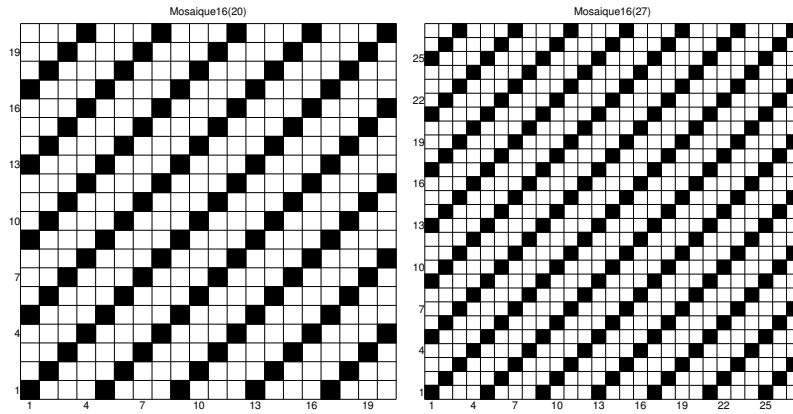
On dispose de la fonction `black(i, j)` qui dessine un pavé noir en ligne i et colonne j d'un quadrillage. Voici le résultat de la commande `black(1,2)` sur le quadrillage précédent :



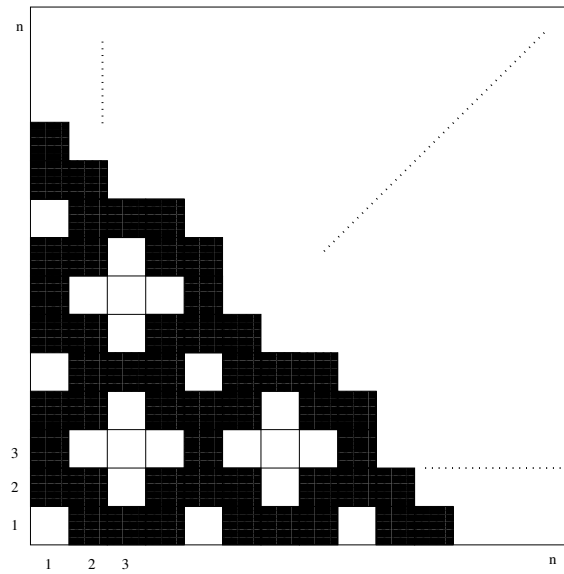
Q. 2 Ecrire la fonction *Mosaïque16(n)* permettant de créer la mosaïque suivante sur le quadrillage *Quadrillage(1, n, 1, n)*



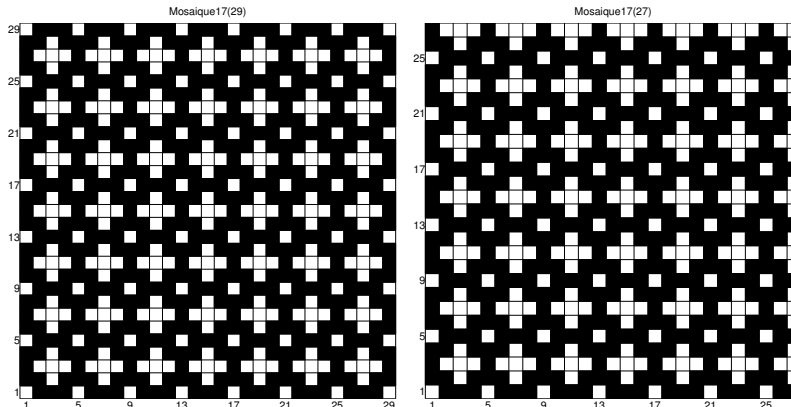
sachant que la diagonale reliant les positions $(1,1)$ et (n,n) est noire. Voici deux exemples :



Q. 3 Ecrire la fonction $Mozaique17(n)$ permettant de créer la mozaïque suivante sur le quadrillage $Quadrillage(1, n, 1, n)$



sachant que le carré en position (1,1) est blanc. Voici deux exemples :



A faire avant 11h00

- ◇ Créer une archive compressée nommée <NOM>-TP1-Q1a3 contenant les fichiers Quadrillage.m, black.m, Mosaïque16.m, Mosaïque17.m et tout autre fichier permettant l'exécution de Mosaïque16.m et Mosaïque17.m. Ici <NOM> correspond évidemment à votre nom.
- ◇ Envoyer un mail à cuvelier@math.univ-paris13.fr ayant pour **objet** "<NOM> TP1 Q1a3" et en fichier joint l'archive compressée créée précédemment.

2 Résolution numérique d'équations différentielles ordinaires

Pour une explication détaillée voir le polycopié fourni CoursED0.pdf

2.1 Rappels de schémas numériques usuels

Définition 2.1 (problème de Cauchy) Soit f l'application continue définie par

$$\begin{aligned} \mathbf{f} : [t^0, t^0 + T] \times \mathbb{R}^d &\longrightarrow \mathbb{R}^d \\ (t, \mathbf{y}) &\longmapsto \mathbf{f}(t, \mathbf{y}) \end{aligned}$$

avec $T \in]0, +\infty]$. Le **problème de Cauchy** revient à chercher une fonction \mathbf{y} définie par

$$\begin{aligned} \mathbf{y} : [t^0, t^0 + T] &\longrightarrow \mathbb{R}^d \\ t &\longmapsto \mathbf{y}(t) \end{aligned}$$

continue et dérivable, telle que

$$\mathbf{y}'(t) = \mathbf{f}(t, \mathbf{y}(t)), \quad \forall t \in [t^0, t^0 + T] \tag{2.1}$$

$$\mathbf{y}(t^0) = \mathbf{y}^{[0]} \in \mathbb{R}^d. \tag{2.2}$$

On note $t^n, n \in \llbracket 0, N \rrbracket$, une discrétisation régulière de $[t^0, t^0 + T]$, $\mathbf{y}^{[n]} \approx \mathbf{y}(t^n)$ et $\mathbf{f}^{[n]} = \mathbf{f}(t^n, \mathbf{y}^{[n]})$

2.1.1 Schéma d'Euler progressif

$$\begin{cases} \mathbf{y}^{[n+1]} &= \mathbf{y}^{[n]} + h\mathbf{f}^{[n]}, \quad \forall n \in \llbracket 0, N-1 \rrbracket \\ \mathbf{y}^{[0]} &= \mathbf{y}(t^0) \end{cases} \tag{2.3}$$

Ce schéma est d'ordre 1.

2.1.2 Schéma de la tangente améliorée

$$\begin{cases} \mathbf{y}^{[n+1]} &= \mathbf{y}^{[n]} + h\mathbf{f}(t^n + \frac{h}{2}, \mathbf{y}^{[n]} + \frac{h}{2}\mathbf{f}^{[n]}), \forall n \in \llbracket 0, N-1 \rrbracket \\ \mathbf{y}^{[0]} &= \mathbf{y}(t^0) \end{cases} \quad (2.4)$$

Ce schéma est d'ordre 2.

2.1.3 Schéma de Runge-Kutta d'ordre 4

$$\begin{aligned} \mathbf{k}_1^{[n]} &= \mathbf{f}(t^n, \mathbf{y}^{[n]}) \\ \mathbf{k}_2^{[n]} &= \mathbf{f}(t^n + \frac{h}{2}, \mathbf{y}^{[n]} + \frac{h}{2}\mathbf{k}_1^{[n]}) \\ \mathbf{k}_3^{[n]} &= \mathbf{f}(t^n + \frac{h}{2}, \mathbf{y}^{[n]} + \frac{h}{2}\mathbf{k}_2^{[n]}) \\ \mathbf{k}_4^{[n]} &= \mathbf{f}(t^n + h, \mathbf{y}^{[n]} + h\mathbf{k}_3^{[n]}) \\ \mathbf{y}^{[n+1]} &= \mathbf{y}^{[n]} + \frac{h}{6}(\mathbf{k}_1^{[n]} + 2\mathbf{k}_2^{[n]} + 2\mathbf{k}_3^{[n]} + \mathbf{k}_4^{[n]}). \end{aligned} \quad (2.5)$$

2.1.4 Méthodes d'Adams-Bashforth

On note $\mathbf{f}^{[n]} = \mathbf{f}(t^n, \mathbf{y}^{[n]})$

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n]} + \frac{h}{2} (3\mathbf{f}^{[n]} - \mathbf{f}^{[n-1]}). \quad (2.6)$$

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n]} + \frac{h}{12} (23\mathbf{f}^{[n]} - 16\mathbf{f}^{[n-1]} + 5\mathbf{f}^{[n-2]}). \quad (2.7)$$

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n]} + \frac{h}{24} (55\mathbf{f}^{[n]} - 59\mathbf{f}^{[n-1]} + 37\mathbf{f}^{[n-2]} - 9\mathbf{f}^{[n-3]}). \quad (2.8)$$

Ces schémas sont **explicites** et leur ordre correspond au nombre de pas.

2.1.5 Méthodes d'Adams-Moulton

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n]} + \frac{h}{2} (\mathbf{f}^{[n+1]} + \mathbf{f}^{[n]}). \quad (2.9)$$

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n]} + \frac{h}{12} (5\mathbf{f}^{[n+1]} + 8\mathbf{f}^{[n]} - \mathbf{f}^{[n-1]}). \quad (2.10)$$

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n]} + \frac{h}{24} (9\mathbf{f}^{[n+1]} + 19\mathbf{f}^{[n]} - 5\mathbf{f}^{[n-1]} + \mathbf{f}^{[n-2]}). \quad (2.11)$$

Ces schémas sont **implicites** et leur ordre correspond au nombre de pas plus un.

2.2 Schéma prédicteur-correcteur

Il s'agit là d'une des méthodes les plus employées. Une méthode de prédiction-correction procède en deux temps : on fournit explicitement une valeur approchée de la solution au $n^{ième}$ pas (soit $\bar{\mathbf{y}}^{[n+1]}$), puis on calcule la valeur correspondante de \mathbf{f} (soit $\bar{\mathbf{f}}^{[n+1]}$) et enfin, on substitue cette valeur dans un schéma implicite (on obtient alors une valeur *corrigée*).

pour n variant de 0 à $N-1$ faire
 Calculer une valeur approchée $\bar{\mathbf{y}}^{[n+1]}$ par un schéma explicite ;
 Evaluer $\bar{\mathbf{f}}^{[n+1]} = \mathbf{f}(t^{n+1}, \bar{\mathbf{y}}^{[n+1]})$;
 $\mathbf{y}^{[n+1]}$ à l'aide d'un schéma implicite en remplaçant l'inconnue par $\bar{\mathbf{y}}^{[n+1]}$;
 finpour

2.3 Travail à effectuer

Le but est de représenter graphiquement les erreurs données par plusieurs schémas et de retrouver numériquement leur ordre. Pour cela il faudra pouvoir connaître explicitement la solution du problème de Cauchy étudié. Voir l'annexe 4.1 pour plusieurs exemples de problèmes de Cauchy avec solutions.

Q. 4 1. Ecrire les cinq fonctions Matlab suivantes correspondant à la résolution d'un problème de Cauchy :

- `[t, Y]=redEUP(f, a, b, yo, N)` : schéma d'Euler progressif (fichier `redEUP.m`).
- `[t, Y]=redTGA(f, a, b, yo, N)` : schéma de la tangente améliorée (fichier `redTGA.m`).
- `[t, Y]=redRK4(f, a, b, yo, N)` : schéma de Runge et Kutta d'ordre 4 (fichier `redRK4.m`).
- `[t, Y]=redAB4(f, a, b, yo, N)` : schéma d'Adams-Bashforth d'ordre 4 (fichier `redAB4.m`).
- `[t, Y]=redPC3(f, a, b, yo, N)` : schéma de type prédiction-correction utilisant les schémas d'Adams-Bashforth d'ordre 3 et d'Adams-Moulton d'ordre 3 (fichier `redPC3.m`).

Ici les paramètres f, a, b, yo correspondent respectivement aux $f, t^0, t^0 + T, \mathbf{y}^{[0]}$ du problème de Cauchy (2.1-2.2). Enfin, Y est le tableau contenant les $\mathbf{y}^{[n]}$, $n \in \{0, \dots, N\}$ et t est le tableau contenant les $n+1$ nombres t^n , $n \in \{0, \dots, N\}$

2. Ecrire le programme principal (fichier `erreur.m`) permettant le calcul et le tracé des erreurs. Pour une méthode donnée le tracé de l'erreur correspond au tracé de l'ensemble des points $(t^n, \text{abs}(\mathbf{y}^{[n]} - \mathbf{y}(t^n)))$, $n \in \{0, \dots, N\}$.

Voir la figure 1 pour un exemple de tracé.

3. Ecrire le programme principal (fichier `ordre.m`) permettant de calculer numériquement l'ordre des 5 schémas et de le représenter.

Voir la figure 2 pour un exemple de tracé. Pour cette figure, la commande Matlab `subplot` a été utilisée (voir Annexe ??) Pour tracer dans l'axe I , on utilise alors la commande `subplot(M, N, I)` et ensuite toutes les commandes graphiques s'appliqueront à l'axe I sélectionné.

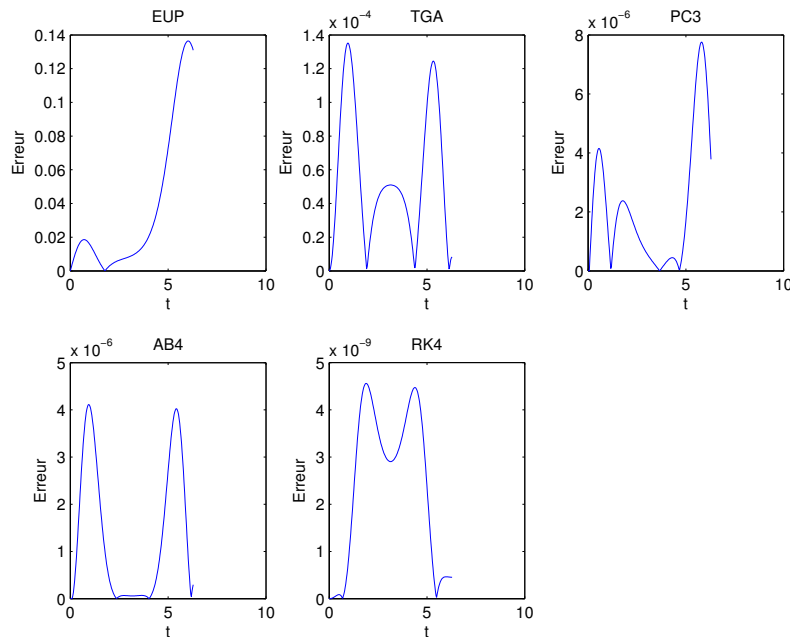


FIGURE 1 – Valeurs absolues des erreurs des 5 schémas

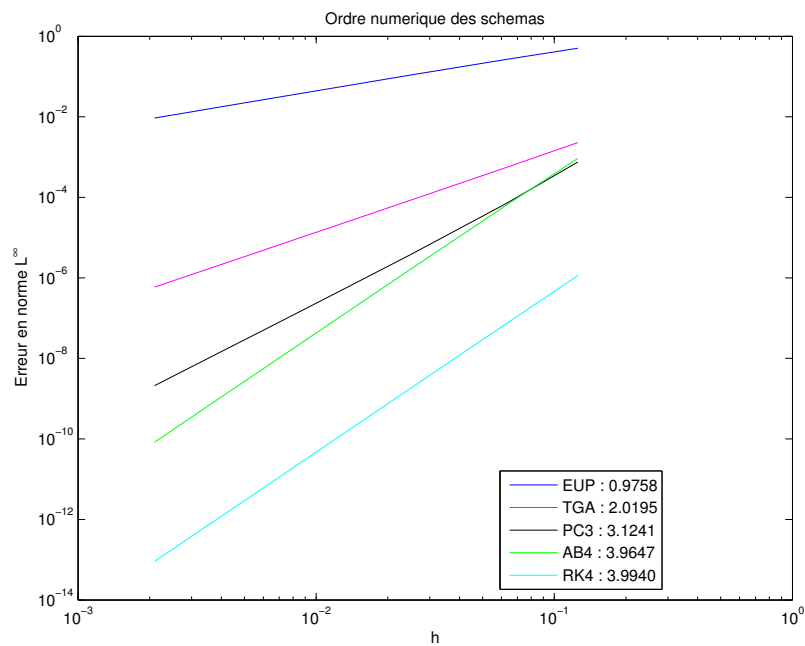


FIGURE 2 – Ordre des 5 schémas

A faire avant 14h30

- ◇ Créer une archive compressée nommée <NOM>-TP1-Q4 contenant les fichiers `redEUP.m`, `redTGA.m`, `redAB4.m`, `redRK4.m`, `redPC3.m`, `erreur.m` et `ordre.m`. Ici <NOM> correspond évidemment à votre nom.
- ◇ Envoyer un mail à `cuvelier@math.univ-paris13.fr` ayant pour **sujet** "<NOM> TP1 Q4" et en fichier joint l'archive compressée créée précédemment.

3 Le système solaire

3.1 Position du problème et équations différentielles

Le but de cette partie est de prédire la position de planètes dans le système solaire à partir de positions et de vitesses initiales des planètes. Les unités choisies sont : les masses sont relatives au soleil, les distances sont en unités astronomiques (1 U.A. = 149 597 870 km) et le temps en jour terrestre.

En astrophysique, la masse solaire est l'unité de masse conventionnellement utilisée pour exprimer les masses des corps célestes massifs et des structures formées d'étoiles (amas, superamas, galaxies, etc.). Son symbole et sa valeur sont :

$$M_{\odot} = 1,9891 \times 10^{30} \text{ kg.}$$

La masse solaire peut aussi être déterminé par l'Unité astronomique (U.A.), l'année et le Constante gravitationnelle (G) :

$$M_{\odot} = \frac{4\pi^2 \times (1\text{UA})^3}{G \times (1\text{ans})^2}.$$

La constante gravitationnelle est alors

$$G = \frac{4\pi^2}{(365,25636567)^2} \approx 2.959130713485796 \times 10^{-4},$$

une année sidérale correspondant à 365,25636567 jours.

On note $\mathbf{q}_i(t)$ la position du i ème corps au temps t , $\mathbf{v}_i(t)$ sa vitesse au temps t et m_i sa masse. Les données initiales pour le soleil (corps $i = 1$) sont $\mathbf{q}_1 = (0, 0, 0)^t$ et $\mathbf{v}_1 = (0, 0, 0)^t$. Toutes les autres données sont fournies dans [?] page 14, regroupées dans le fichier `SysSolDataHairer.m` et correspondent aux positions de différentes planètes le 5 septembre 1994 à 0h00m00s. D'autres données, récupérées sur JPL Solar System Dynamics sont disponibles dans le fichier `SysSolData.m` et correspondent aux positions de différentes planètes le 14 avril 2015 à 0h00m00s.

De manière classique, un problème à N corps se modélise par

$$\frac{d^2\mathbf{q}_i}{dt^2}(t) = -G \sum_{j=1, j \neq i}^N m_j \frac{\mathbf{q}_i(t) - \mathbf{q}_j(t)}{\|\mathbf{q}_i(t) - \mathbf{q}_j(t)\|^3}, \quad \forall i \in \llbracket 1, N \rrbracket.$$

et l'énergie du système ,correspondant à la somme des énergies cinétiques et des énergies potentielles gravitationnelle (voir wikipedia), est donnée par

$$\mathcal{E}(t) = \frac{1}{2} \sum_{i=1}^N m_i \|\mathbf{v}_i(t)\|^2 - G \sum_{i=1}^N \sum_{j=1}^{i-1} \frac{m_i m_j}{\|\mathbf{q}_i(t) - \mathbf{q}_j(t)\|}.$$

3.2 Résolution numérique classique

Q. 5 (Sur feuille) 1. *Ecrire de manière détaillée le problème de Cauchy associé à un problème à N corps sachant que la fonction vectorielle inconnues \mathbf{y} est définie par*

$$\mathbf{y}(t) = \begin{pmatrix} \mathbf{q}_1(t) \\ \vdots \\ \mathbf{q}_N(t) \\ \mathbf{q}'_1(t) \\ \vdots \\ \mathbf{q}'_N(t) \end{pmatrix} \in \mathbb{R}^{6N}$$

2. *Quelles sont les données du problème de Cauchy obtenu ? (avec leur type détaillé : entier, réel, complexe, vecteur, matrice, fonction, ...)*
3. *Quelles sont les inconnues du problème de Cauchy obtenu ? (avec leur type détaillé : entier, réel, complexe, vecteur, matrice, fonction, ...)* ■

3.3 Système à 3 corps : Soleil, Saturne et Jupiter

Pour récupérer les données initiales de ce système, on peut utiliser les commandes Matlab :

```
SysSol3=SysSolData();  
SysSol3.planetes=SysSol3.planetes(1:3);
```

Les deux fonctions fournies `plotSysSol` et `PlotPlanets` (necessite `PlotPlanet`) permettent de représenter, après calculs, respectivement l'orbite des planetes de la structure `SysSol` (voir fonctions `SysSolData` ou `SysSolDataHairer`) et les planètes à un instant donné. Leur syntaxe est la suivante `plotSysSol(T,Y,SysSol)` et `PlotPlanets(SysSol,q)` avec

- (T,Y) tableaux retournés lors de la résolution du problème de Cauchy associé,
- `SysSol` structure contenant divers renseignements sur le système solaire,
- `q` positions des planetes.

- Q. 6 (Matlab)**
1. Ecrire la fonction Matlab `fSysSol3` (fichier `fSysSol3.m`) correspondant à la fonction `f` du problème de Cauchy associé au problème à 3 corps (Soleil, Saturne et Jupiter). On pourra utiliser une variable globale pour les paramètres physiques (voir fichiers `SysSolDataHairer.m` ou `SysSolData.m`).
 2. Ecrire le programme `prgSysSol3` (fichier `prgSysSol3.m`) qui résoud le problème de Cauchy par `resEUP` et `resRK4` puis représente, pour chaque méthode, les orbites des 3 planètes ainsi que leurs positions à l'instant final avec comme données $T = 24000$ (jours) et $N = 8000$ (voir figure 3) On représentera aussi l'énergie du système pour chaque méthode.

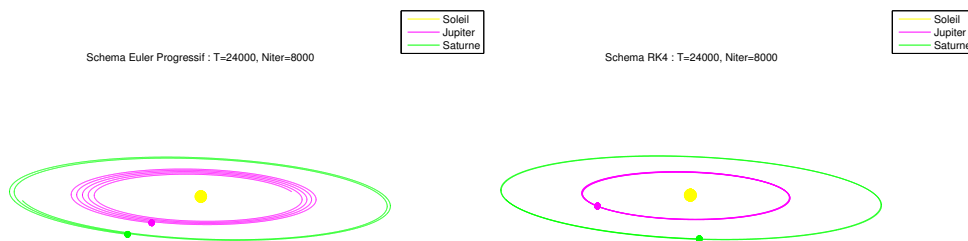


FIGURE 3 – Orbites Soleil-Saturne-Jupiter

A faire avant 16h30

- ◊ Rendre la réponse à la question 5. N'oubliez pas d'écrire votre nom sur les documents rendus!
- ◊ Créer une archive compressée nommée `<NOM>-TP1-Q6` contenant l'ensemble des fichiers nécessaire à l'exécution de `prgSysSol3`. Ici `<NOM>` correspond évidemment à votre nom.
- ◊ Envoyer un mail à `cuvelier@math.univ-paris13.fr` ayant pour **objet** "`<NOM> TP1 Q6`" et en fichier joint l'archive compressée créée précédemment.

3.4 Système à 6 corps

On étudie ici le système composé des 6 planetes Soleil, Saturne, Jupiter, Uranus, Neptune et Pluton.

- Q. 7 (Matlab)**
1. Ecrire la fonction Matlab `fSysSol6` (fichier `fSysSol6.m`) correspondant à la fonction `f` du problème de Cauchy associé au problème à 6 corps (Soleil, Saturne, Jupiter, Uranus, Neptune et Pluton). On pourra utiliser une variable globale pour les paramètres physiques (voir fichiers `SysSolDataHairer.m` ou `SysSolData.m`).

2. Ecrire le programme `prgSysSol6` (fichier `prgSysSol6.m`) qui résoud le problème de Cauchy par `resEUP` et `resRK4` puis représente, pour chaque méthode, les orbites des 6 planètes ainsi que leurs positions à l'instant final avec comme données $T = 100000$ (jours) et $N = 10000$ (voir figure 4) On représentera aussi l'énergie du système pour chaque méthode.

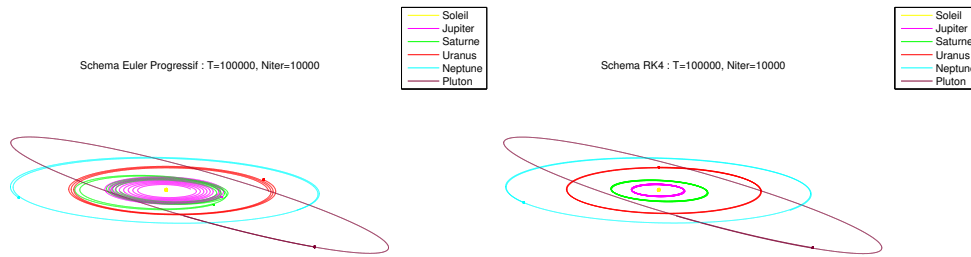


FIGURE 4 – Orbites Soleil, Saturne, Jupiter, Uranus, Neptune et Pluton

A faire avant la fin du TP

- ◇ Créer une archive compressée nommée `<NOM>-TP1-Q7` contenant l'ensemble des fichiers nécessaires à l'exécution de `prgSysSol6`. Ici `<NOM>` correspond évidemment à votre nom.
- ◇ Envoyer un mail à `cuvelier@math.univ-paris13.fr` ayant pour **sujet** "`<NOM> TP1 Q7`" et en fichier joint l'archive compressée créée précédemment.

3.5 Système à 7 corps

On étudie ici le système composé des 7 planetes Soleil, Saturne, Jupiter, Uranus, Neptune, Pluton et Terre.

- Q. 8 (Matlab)**
1. Ecrire la fonction Matlab `fSysSol7` (fichier `fSysSol7.m`) correspondant à la fonction f du problème de Cauchy associé au problème à 7 corps (Soleil, Saturne, Jupiter, Uranus, Neptune, Pluton et Terre). On pourra utiliser une variable globale pour les paramètres physiques (voir fichiers `SysSolDataHairer.m` ou `SysSolData.m`).
 2. Ecrire le programme `prgSysSol7` (fichier `prgSysSol7.m`) qui résoud le problème de Cauchy par `resEUP` et `resRK4` puis représente, pour chaque méthode, les orbites des 7 planètes ainsi que leurs positions à l'instant final avec comme données $T = 100000$ (jours) et $N = 10000$. On représentera aussi l'énergie du système pour chaque méthode. ■

A faire avant la fin du TP

- ◇ Créer une archive compressée nommée `<NOM>-TP1-Q8` contenant l'ensemble des fichiers nécessaire à l'exécution de `prgSysSol7`. Ici `<NOM>` correspond évidemment à votre nom.
- ◇ Envoyer un mail à `cuvelier@math.univ-paris13.fr` ayant pour **sujet** "`<NOM> TP1 Q8`" et en fichier joint l'archive compressée créée précédemment.

4 Annexes

4.1 Quelques E.D.O. du premier ordre

4.1.1 Exemple 1

Soit $\alpha \in \mathbb{R}$. L'E.D.O.

$$\begin{cases} y'(t) &= \cos(t), \forall t \geq 0, \\ y(0) &= \alpha, \end{cases}$$

a pour solution $y(t) = \sin(t) + \alpha$.

4.1.2 Exemple 2

Soit $\beta \in \mathbb{R}$. L'E.D.O.

$$\begin{cases} y'(t) &= \sin(t), \forall t \geq 0, \\ y(0) &= \beta, \end{cases}$$

a pour solution $y(t) = -\cos(t) + 1 + \beta$.

4.1.3 Exemple 3

L'E.D.O.

$$\begin{cases} y'(t) &= -y(t) \sin(t), \forall t \geq 0, \\ y(0) &= e = \exp(1), \end{cases}$$

a pour solution $y(t) = \exp(\cos(t))$.